Amendments to the Claims

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) An object-oriented computer system~~, including~~ comprising:

two or more class loaders for loading program class files into the system; and

a constraint checking mechanism so that where a first class file loaded by a first class loader makes a symbolic reference to a second class file loaded by a second class loader, said symbolic reference including a descriptor of a third class file, the constraint enforces that the first and second class files agree on the identity of the third class file,

said constraint checking mechanism including means for creating a data structure for recording a constraint as an asymmetric relationship between two class loaders, wherein said data structure includes, for a class loader which has loaded a class file that contains a symbolic reference to another class file, a first parameter denoting the class file which is identified by a descriptor in said symbolic reference, and a second parameter denoting the class loader which loaded said another class file.

2. (Original) The system of claim 1, wherein said data structure further includes a third parameter denoting the object reference to said class file which is identified by a descriptor in said symbolic reference, as loaded by the class loader with which the data structure is associated.

3. (Original) The system of claim 2, wherein said data structure further includes a fourth parameter, denoting the object reference to said class file which

is identified by a descriptor in said symbolic reference, as loaded by said class loader which loaded said another class file.

4. (Original) The system of claim 3, further comprising means for comparing said third and fourth parameters, to identify a constraint violation if they do not match.

5. (Currently amended) The system of claim 2, further comprising means for copying said third parameter into a data structure associated with said class loader which ~~loader~~ loaded said another class file.

6. (Original) The system of claim 1, wherein each class loader has its own cache, and the data structure for a class loader is stored in the cache for that class loader.

7. (Currently amended) ~~A method of operating~~ In an object-oriented computer system~~, including~~ a method comprising:

providing two or more class loaders for loading program class files into the system and a constraint checking mechanism so that where a first class file loaded by a first class loader makes a symbolic reference to a second class file loaded by a second class loader, said symbolic reference including a descriptor of a third class file, the constraint enforces that the first and second class files agree on the identity of the third class file~~, said method comprising the steps of:~~;

identifying the need for a constraint between said first and second class loaders in respect of said third class file;

creating a data structure for each of said first and second class loaders; and

setting a pointer from the data structure for the first class loader to the data structure for the second class loader to identify the latter as being the constraint parent.

8. (Original) The method of claim 7, wherein the data structure for each of said first and second class loaders is stored in a cache associated with the respective class loader.

9. (Original) The method of claim 7, further comprising the steps of:

resolving said third class file to a first object reference by the first class loader;

copying the first object reference from the third class file to the data structure for the second class loader;

resolving said third class file to a second object reference by the second class loader; and

checking that said first and second object references are identical to ensure that said constraint has not been violated.

10. (Original) The method of claim 7, further comprising the steps of:

resolving said third class file to a first object reference by the second class loader;

resolving said third class file to a second object reference by the first class loader; and

responsive to detecting that said pointer is set , checking that said second and first class references are identical to ensure that said constraint has not been violated.

11. (Currently amended) ~~A method of operating~~ In an object-oriented computer system~~, including~~ a method comprising:

providing two or more class loaders for loading program class files into the system and a constraint checking mechanism so that where a first class file loaded by a first class loader makes a symbolic reference to a second class file loaded by a second class loader, said symbolic reference including a descriptor of a third class file, a constraint enforces that the first and second class files agree on the identity of the third class file~~, said method comprising the steps of:~~;

providing a data store for asymmetrically recording the constraint between said first and second class loaders in respect of said third class file;

resolving a reference to said third class file by said first class loader;

identifying from the data store the existence of said constraint between said first and second class loaders in respect of said third class file; and

updating the data store to indicate the reference to said third class file as resolved by the first class loader.

12. (Original) The method of claim 11, wherein said data store comprises a first data structure in a cache associated with the first class loader, and a second data structure in a cache associated with the second class loader.

13. (Original) The method of claim 12, wherein the asymmetric recording of a constraint comprises a pointer from said first data structure to said second data structure.

14. (Original) The method of claim 12, wherein said step of updating comprises updating said second data structure with the reference.